UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/747,583	12/21/2000	Daniel Leibholz	SMQ-023	3889

959 7590 07/11/2006

LAHIVE & COCKFIELD
28 STATE STREET
BOSTON, MA 02109

EXAMINER	
HUISMAN, DAVID J	

ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 07/11/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/747,583	LEIBHOLZ ET AL.	
	Examiner	Art Unit	
	David J. Huisman	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *23 March 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-19* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-19* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *23 April 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-19 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: Amendment as received on 3/23/2006.

Claim Objections

3. Regarding claim 4, the examiner would like applicant to confirm that line 2 should in fact say "the registers" instead of --the storage--. It appears as if it should be the latter.

4. Claim 7 recites the limitation "the stack" in line 3. There is insufficient antecedent basis for this limitation in the claim.

5. Claim 14 is objected to because of the following informalities: Please insert --window-- after "underflow". Also, in the last paragraph, replace "or" with --and--. Appropriate correction is required.

6. Claim 16 is objected to because of the following informalities: In the 2nd paragraph, please insert --window--after "underflow". Also, in the 2nd to last line of the claim, replace "or" with --and--. Appropriate correction is required.

Claim Rejections - 35 USC § 112

7. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it

> pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

8. Claims 14-15 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the enablement requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to enable one skilled in the art to which it pertains, or with which it is most nearly connected, to make and/or use the invention. Regarding claim 14, the examiner does not believe that one of ordinary skill in the art would be able to make/use a system, without undue experimentation, that has a detector for detecting an instruction in a cache by determining if execution of any fetched instruction will result in overflow or underflow. That is, it is not clear how determining if execution of a second instruction, for instance, will result in overflow/underflow allows one to detect a first instruction in the cache. Does applicant mean to say "said instruction" instead of "any fetched instructions" and if so, how does execution of "said instruction" allow a detector to detect said instruction in a cache? They don't seem to be related.

Claim Rejections - 35 USC § 102

9. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

10. Claims 1-13 and 16-19 are rejected under 35 U.S.C. 102(e) as being anticipated by Lin U.S. Patent Number 6,631,452 (as applied in the previous Office Action).

11. Referring to claim 1, Lin has taught a microprocessor, comprising:

a) registers for holding values, wherein said registers are logically partitioned into register windows. See Fig.2, Fig.4, and column 4, lines 1-29, and note that the frames are the register windows.

b) a storage for storing values held in the registers of the register windows. See Fig.2, Fig.4, and column 4, lines 1-29, and note the backing store.

c) a detector for detecting that a register window overflow condition or a register window underflow condition is imminent, by determining if execution of any fetched instructions will result in a register window overflow condition or a register underflow condition, and an instruction generator responsive to the detector for generating at least one instruction to manipulate the storage to avoid a trap to avoid stalling the microprocessor, wherein the trap performs at least one of a register window spill operation or a register window fill operation responsive to the condition that is detected as imminent. See column 2, line 47, to column 3, line 5, column 3, lines 59-67, and column 5, lines 26-41, and note that speculative spills/fills are performed when bandwidth is available. The speculative spills/fills are performed because the system anticipates that overflow/underflow is imminent due to an instruction that, when fetched, will require more storage than is available to a procedure or due an instruction that, when fetched, will access data that is no longer in the register file assigned to a procedure. See column 5, lines 26-41. The successful performance of speculative spills/fills eliminates the need for performing mandatory spills/fills. See column 5, line 42, to column 6, line 2. In the case when a speculative spill/fill is successful, the mandatory spill/fill would be unnecessary as the speculative spill/fill has already completed the task to be performed by the mandatory spill/fill. This prevents the stall associated with the mandatory spill/fill from being incurred. When a

speculative spill/fill is successful, then the associated mandatory spill/fill is avoided. This is clearly Lin's intention as the purpose of Lin's system is to reduce stalls associated with mandatory spills/fills.

12. Referring to claim 2, Lin has taught a microprocessor as described in claim 1. Lin has further taught that the detector and the instruction generator are implemented in hardware. See column 11, lines 1-15.

13. Referring to claim 3, Lin has taught a microprocessor as described in claim 1. Lin has further taught that the microprocessor further comprises a cache for caching instructions (Fig.1, component 130) for introduction into an execution stage and wherein the detector examines the instructions in the cache to determine if a register window overflow condition is imminent by determining if execution of any of the fetched instructions will result in a register window overflow condition. It should be realized that by executing fetched instructions of the active procedure (from a cache), the system is detecting that an overflow is imminent, i.e., that the active procedure may need more registers. Therefore, speculative spills will be issued. The instructions in the cache are the same instructions that are to be executed. No matter what point the instructions are monitored, they are "the instructions in the cache". That is, applicant is not claiming that the instructions are monitored while in the cache and before being fetched from the cache for decoding, execution, etc. Applicant merely claims monitoring instructions in the cache and an instruction that is being executed is also an instruction in the cache as that is where it was fetched from.

14. Referring to claim 4, Lin has taught a microprocessor as described in claim 3. Lin has further taught that the detector looks for an instruction in the cache that stores contents of a

register window in the registers when the registers have no available space for storing the contents (Lin column 7 line 45-column 8 line 15, column 5 line 46-column 6 line 2, abstract, figure 6, column 8 line 49-column 9 line 5).

15. Referring to claim 5, Lin has taught a microprocessor as described in claim 3. Lin has further taught that the detector examines how much storage space is available in the registers (Lin column 7 line 45-column 8 line 15, column 5 line 46-column 6 line 2, abstract, figure 6, column 8 line 49-column 9 line 5).

16. Referring to claim 6, Lin has taught a microprocessor as described in claim 1. Lin has further taught that the microprocessor further comprises a cache for caching instructions for introduction into an execution stage and wherein the detector examines the instructions in the cache to determine if a register window underflow condition is imminent by determining if execution of the instructions will result in a register window underflow condition. It should be realized that by executing fetched instructions of the active procedure (from a cache), the system is detecting that an underflow is imminent, i.e. that an inactive procedure will be returned to (and not have the required data available). Therefore, speculative fills will be issued to speculatively load that data. The instructions in the cache are the same instructions that are to be executed. No matter what point the instructions are monitored, they are "the instructions in the cache". That is, applicant is not claiming that the instructions are monitored while in the cache and before being fetched from the cache for decoding, execution, etc. Applicant merely claims monitoring instructions in the cache and an instruction that is being executed is also an instruction in the cache as that is where it was fetched from.

17. Referring to claim 7, Lin has taught a microprocessor as described in claim 6. Lin has further taught that the detector looks for an instruction in the cache that restores a register window when contents of the register window are stored on the stack rather than in the registers (Lin column 7 line 45-column 8 line 15, column 5 line 46-column 6 line 2, abstract, figure 6, column 8 line 49-column 9 line 5).

18. Referring to claim 8, Lin has taught a microprocessor as described in claim 1. Lin has further taught that the detector detects solely whether a register window underflow condition is imminent (Lin column 11 lines 40-43).

19. Referring to claim 9, Lin has taught a microprocessor as described in claim 1. Lin has further taught that the detector detects solely whether a register window overflow condition is imminent (Lin column 11 lines 35-39).

20. Referring to claim 10, Lin has taught a microprocessor as described in claim 1. Lin has further taught that the detector detects both whether a register window overflow condition is imminent and whether a register window underflow condition is imminent. It should be noted that spills and fills are performed in response to detecting overflows and underflows, respectively.

21. Referring to claim 11, Lin has taught a microprocessor as described in claim 1. Lin has further taught that the microprocessor further comprises an execution unit for executing the instruction generated by the instruction generator (Lin column 7 line 45-column 8 line 15, column 5 line 46-column 6 line 2, abstract, figure 4).

22. Referring to claim 12, Lin has taught a microprocessor as described in claim 1. Lin has further taught that the microprocessor performs out of order execution of instructions (Lin

column 3 lines 32-45; speculative processing by definition is not a certain operation, but is a prediction based on some information, and because of a misprediction, the wrong order of operations will occur).

23. Referring to claim 13, Lin has taught a microprocessor as described in claim 1. Lin has further taught that the instruction generator includes a second storage for holding the at least one instruction that is generated by the instruction generator (Lin column 7 line 45-column 8 line 15, column 5 line 46-column 6 line 2, abstract; it is inherent that some ROM or other memory would be available to hold the speculative operations, that the RSE chooses to executes).

24. Referring to claim 16 Lin has taught a microprocessor having a plurality of registers logically partitioned into register windows and a storage for storing contents of register windows, a method, comprising the steps of:

a) determining that a register window overflow condition or a register window underflow condition is imminent by performing a logic operation on a value representative of a state of a register and a value representative of an instruction held in cache and determining if execution of any fetched instructions will result in a register window overflow condition or a register underflow condition, and in response to determining that the register overflow condition or the register window underflow condition is imminent, manipulating the storage to avoid a trap performing at least one of a register window spill operation or a register window fill operation responsive to the condition determined as imminent.. See column 2, line 47, to column 3, line 5, column 3, lines 59-67, and column 5, lines 26-41, and note that speculative spills/fills are performed when bandwidth is available. The speculative spills/fills are performed because the system anticipates that overflow/underflow is imminent due to an instruction that, when fetched,

will require more storage than is available to a procedure or due an instruction that, when fetched, will access data that is no longer in the register file assigned to a procedure. See column 5, lines 26-41. The successful performance of speculative spills/fills eliminates the need for performing mandatory spills/fills. See column 5, line 42, to column 6, line 2. In the case when a speculative spill/fill is successful, the mandatory spill/fill would be unnecessary as the speculative spill/fill has already completed the task to be performed by the mandatory spill/fill. This prevents the stall associated with the mandatory spill/fill from being incurred. When a speculative spill/fill is successful, then the associated mandatory spill/fill is avoided. This is clearly Lin's intention as the purpose of Lin's system is to reduce stalls associated with mandatory spills/fills. Also, see column 6, lines 16-23, and Table 2 in column 8. The value representative of a state of a register would be the RSE.StoreReg, which holds the next dirty register to be spilled. And, the value representative of a fetched instruction would be the RSC.mode value. This value indicates, for the instruction, whether or not speculative mode is used. A logic operation is performed on these values. The logic operation is a compare of the type "if a dirty register exists in RSE.StoreReg, and speculative mode is enabled, then perform a speculative spill when possible." As long as there is a value in StoreReg which should be spilled, then overflow is imminent. That is, data may need to use this register when a new procedure is switched in, but the register cannot be used until it is spilled. So, the presence of a dirty register and the fact that instructions from procedures are to be fetched and executed indicates that underflow/overflow is imminent.

25. Referring to claim 17, Lin has taught a method as described in claim 16. Lin has further taught that when it determined that a register window overflow condition is imminent, the step of

manipulating the storage comprises providing at least one instruction for execution by the microprocessor that causes the contents in at least the selected register window to be stored in the storage (Lin column 7 line 45-column 8 line 15, column 5 line 46-column 6 line 2, abstract).

26. Referring to claim 18, Lin has taught a method as described in claim 16. Lin has further taught that when it is determined that a register window underflow condition is imminent, the step of manipulating the storage comprises providing at least one instruction for execution by the microprocessor that causes data in the storage to be stored in the registers (Lin column 7 line 45-column 8 line 15, column 5 line 46-column 6 line 2, abstract).

27. Referring to claim 19, Lin has taught a method as described in claim 16. Lin has further taught that the microprocessor has an instruction stream slated for execution and wherein the instruction that causes the contents in at least the selected register window to be stored in the storage is inserted into the instruction stream (Lin column 7 line 45-column 8 line 15, column 5 line 46-column 6 line 2, abstract, figure 4; the instruction stream is the operations in figure 4 that spill and fill the registers from memory).

28. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

29. Claims 1, 2, 6, 8, and 11-15 are rejected under 35 U.S.C. 102(b) as being anticipated by Eickemeyer et al., U.S. Patent No. 5,377,336 (herein referred to as Eickemeyer).

30. Referring to claim 1, Eickemeyer has taught a microprocessor, comprising:

a) registers for holding values, wherein said registers are logically partitioned into register windows. See Fig.2, and note the register file. A register file has multiple registers that are logically partitioned (separate logic for R1, R2, etc.). And, each register is a window of 1 register.

b) a storage for storing values held in the registers of the register windows. See Fig.2, component 104.

c) a detector for detecting that a register window overflow condition or a register window underflow condition is imminent, by determining if execution of any fetched instructions will result in a register window overflow condition or a register underflow condition, and an instruction generator responsive to the detector for generating at least one instruction to manipulate the storage to avoid a trap to avoid stalling the microprocessor, wherein the trap performs at least one of a register window spill operation or a register window fill operation responsive to the condition that is detected as imminent. See column 3, lines 30-47, and claim 1. Note the system scans for load instructions prior to their execution, where loads require an access to storage to manage register window information. When a load is detected, the system determines that execution of the fetched load will result in register window underflow, which in Eickemeyer would be when the data to be loaded into the register is not in the cache (and so, the system would have to perform a load from main memory (trap)). Also, see the abstract, column 3, lines 30-47, and claim 1. Note that in response to detecting the load, a prefetch instruction is generated in order to instruct the system to perform a prefetch operation. As a result, the system prefetches data which is ultimately written to a register. Hence, a prefetch is an operation associated with filling registers, i.e., it is a register fill operation.

31. Referring to claim 2, Eickemeyer has taught a microprocessor as described in claim 1. Eickemeyer has further taught that the detector and the instruction generator are implemented in hardware. See Fig.1 and Fig.2.

32. Referring to claim 6, Eickemeyer has taught a microprocessor as described in claim 1. Eickemeyer has further taught that the microprocessor further comprises a cache for caching instructions for introduction into an execution stage and wherein the detector examines the instructions in the cache to determine if a register window underflow condition is imminent by determining if execution of the instructions will result in a register window underflow condition. See column 3, lines 30-47, and claim 1. Note the system scans for load instructions prior to their execution, where loads require an access to storage to manage register window information. Note that the buffer of Eickemeyer is essentially a cache by definition. A cache, as is known, is a memory for holding recently accessed data items. In Eickemeyer, instructions are fetched into the buffer, so the buffer is holding recently accessed data items, and is therefore acting as a cache. When a load is detected, the system determines that execution of the fetched load will result in register window underflow, which in Eickemeyer would be when the data to be loaded into the register is not in the cache (and so, the system would have to perform a load from main memory).

33. Referring to claim 8, Eickemeyer has taught a microprocessor as described in claim 1. Eickemeyer has further taught that the detector detects solely whether a register window underflow condition is imminent for reasons described above.

34. Referring to claim 11, Eickemeyer has taught a microprocessor as described in claim 1. Eickemeyer has further taught that the microprocessor further comprises an execution unit for

executing the instruction generated by the instruction generator. There must inherently be logic which executes a command.

35. Referring to claim 12, Eickemeyer has taught a microprocessor as described in claim 1. Eickemeyer has further taught that the microprocessor performs out of order execution of instructions. See column 11, lines 25-29.

36. Referring to claim 13, Eickemeyer has taught a microprocessor as described in claim 1. Eickemeyer has further taught that the instruction generator includes a second storage for holding the at least one instruction that is generated by the instruction generator. See Fig.2, component 210.

37. Referring to claim 14, Eickemeyer has taught in a microprocessor having a storage and registers, an engine, comprising:

a) a detector for detecting an instruction in a cache prior to execution of said instruction indicating that a trap requiring an access to the storage to manage register window information is imminent, by determining if execution of any fetched instructions will result in a register window overflow or a register window underflow. See column 3, lines 30-47, and claim 1. Note the system scans for load instructions prior to their execution, where loads require an access to storage to manage register window information. Note that the buffer of Eickemeyer is essentially a cache by definition. A cache, as is known, is a memory for holding recently accessed data items. In Eickemeyer, instructions are fetched into the buffer, so the buffer is holding recently accessed data items, and is therefore acting as a cache. When a load is detected, the system determines that execution of the fetched load will result in register window

underflow, which in Eickemeyer would be when the data to be loaded into the register is not in the cache (and so, the system would have to perform a load from main memory).

b) an instruction generator responsive to the detector for generating at least one instruction to avoid the trap performing at least one of a register window spill operation or a register window fill operation. See the abstract, column 3, lines 30-47, and claim 1. Note that in response to detecting the load, a prefetch instruction is generated in order to instruct the system to perform a prefetch operation. As a result, the system prefetches data which is ultimately written to a register. Hence, a prefetch is an operation associated with filling registers, i.e., it is a register fill operation.

38. Referring to claim 15, Eickemeyer has taught a microprocessor as described in claim 14. Eickemeyer has further taught that the engine is implemented in hardware. See claim 1, Fig.1, and Fig.2.

Response to Arguments

39. Applicant's arguments filed on March 23, 2006, have been fully considered but they are not persuasive.

40. Applicant argues the novelty/rejection of claim 14 on page 9 of the remarks, in substance that:

> "Eickemeyer does not determine if execution of any fetched instructions will result in a register window overflow condition or a register underflow condition."
>
> "Further, the detector of Eickemeyer detects load instructions in an instruction buffer...A buffer is not equivalent to a cache. While both are memory, they are different in structure function and operation."

41. These arguments are not found persuasive for the following reasons:

a) Regarding the first argument, since applicant has not explicitly defined register window overflow/underflow conditions, the examiner is to assign the broadest, reasonable interpretation to the conditions. In general underflow is when the data you're looking for is not available, and overflow is when you have no room to store data that you want to store. So, Eickemeyer detects a load and realizes that if a prefetch is not performed, that load will cause an underflow, i.e., the data that is to be loaded will not be available in the cache, and so it will need to be retrieved from main memory, which requires more time. Since this data will ultimately be stored in the register window, it is considered to be a register window underflow condition.

b) Regarding the second argument, the buffer of Eickemeyer is essentially a cache by definition. A cache, as is known, is a memory for holding recently accessed data items. In Eickemeyer, instructions are fetched into the buffer, so the buffer is holding recently accessed data items, and is therefore acting as a cache. As an alternative interpretation, components 103 and 105 may be considered an overall cache component.

42. Applicant's arguments with respect to Lin have been considered but are mostly moot because of the new interpretation of Lin in order to address the new claim limitations. A brief summary of Lin, as used in the rejection, is as follows: Lin attempts to avoid mandatory spill/fill operations, and corresponding processor stalls, by opportunistically executing spills/fills when additional bandwidth is available. These opportunistic fills/spills are executed because the system anticipates that underflow/overflow is imminent, i.e., a fetched instruction will soon execute and cause underflow/overflow to occur. Clearly, the system knows that execution of at

least one fetched instruction will cause underflow/overflow (as this is the reason it operates as it does), and so it tries to prevent stalling with the opportunistic spills/fills.

Conclusion

43. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL.** See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (571) 272-4168. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

DJH
David J. Huisman
May 31, 2006

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100